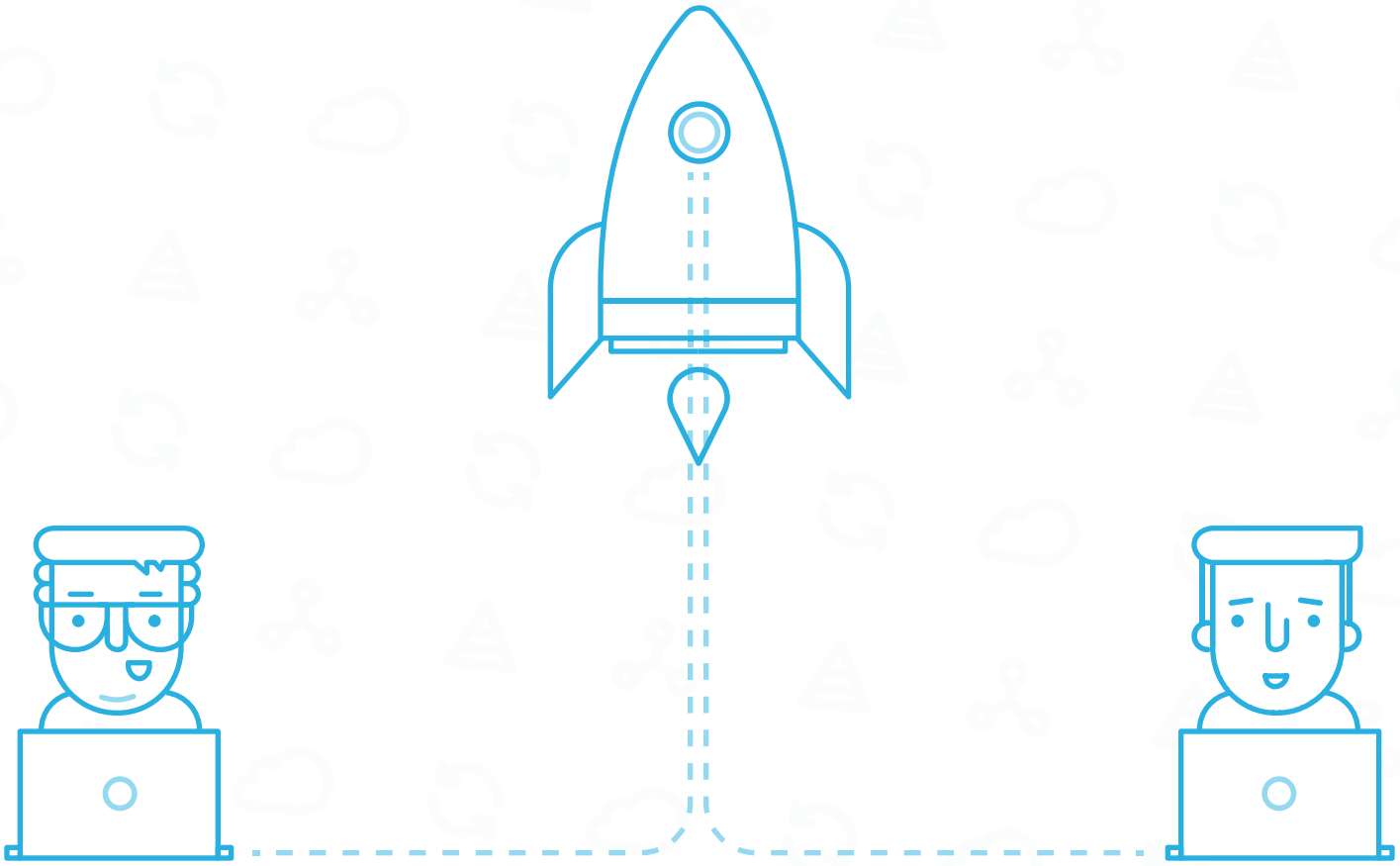
CONTINO

UNBLOCKING THE ENVIRONMENTS LOGJAM WITH A DEVOPS APPROACH

White Paper



Having worked with many enterprise organisations on their DevOps initiatives, the biggest pain point and source of wastage that we see across the software development lifecycle is around environment provisioning, access and management.

This was initially surprising to us. Of all of the challenges associated with development, testing and software operations, access to environments feels as though it should be a tractable problem in todays virtualized, cloud, configuration managed world.

Having looked closer however, we now see why this situation persists:

- Many of the costs associated with what we call The Environment Logjam are hidden – small but frequent delays, costs, inefficiencies and quality implications which are not accounted for and made visible. In isolation these seem trivial and ad-hoc, but in aggregate they are significant.

- The problem is not trivial to solve – requiring engineering effort and investment in automation, platforms, tools and new engineering approaches. This is especially true within enterprises that have diverse, complex, integrated applications that were put into place before virtualization and cloud matured as platforms.

- The combination of these two points means that new features and business as usual takes priority over taking on improvement initiatives in this area. Inefficiencies related to environments become accepted as the norm, impacting the organisations ability to deliver and innovate.

THIS PAPER IS AIMED AT SENIOR IT DECISION MAKERS WHO ARE INTERESTED IN REDUCING THEIR DELIVERY CYCLE TIME AND IMPROVING EFFICIENCIES IN CUSTOM SOFTWARE DELIVERY. THE AIM OF THIS PAPER IS THREEFOLD:

- TO ILLUSTRATE THE HIDDEN COSTS AND INEFFICIENCIES ASSOCIATED WITH WHAT WE CALL THE ENVIRONMENT LOGJAM SO THAT READERS CAN SEE IF THESE PROBLEMS ARE RELEVANT OR PERHAPS ACCEPTED AS THE NORM WITHIN THEIR OWN ORGANISATIONS.

- TO EXPLAIN HOW A DEVOPS APPROACH CAN HELP TO IMPROVE THIS SITUATION – PRIMARILY THROUGH AUTOMATION BUT ALSO CULTURE AND WORKING PRACTICES.

- TO HIGHLIGHT THE BUSINESS CASE FOR TAKING ON IMPROVEMENT INITIATIVES IN THIS AREA.

THE ENVIRONMENT LOGJAM

Below are some of the environment provisioning and management problems that we have frequently observed within enterprise organisations:

ENVIRONMENTS ACT AS A BOTTLENECK TO FREQUENT AGILE SOFTWARE RELEASES

Valuable software that is ready to release can literally left waiting for an environment to be deployed into, and if for any reason a software release misses it's pre-planned slot, the resulting delay waiting for the next slot may be significant.

This might have worked in a waterfall world with carefully planned releases every 3-6 months, but in a dynamic, faster moving agile world with much more frequent releases, it can become a significant source of wastage and inefficiency.

ENVIRONMENT ACCESS IMPACTS PROJECT DELIVERY

People in the IT organization often negotiate over access to environments based on perceived business value of their releases. Indeed, there is sometimes an element of politics or 'pulling rank' to secure access to the value commodity of environment time!

If a high priority release is ready for testing, another release might be delayed or pushed down the queue with significant knock on effects for that program of work.

Likewise, if a testing phase is running late, rather than vacating the environment, releases tend to sit there hogging environments regardless of impact to other projects. This is because the cost of reconfiguring environments is so high that it is usually worth the collateral damage.

The net result of this is unpredictable project delivery lifecycles and projects that run over time and over budget.

ENVIRONMENT MANAGEMENT IS A MANUAL, COMPLEX EXERCISE

The full time role of the Environment Manager is sometimes created within organisations in order to manage access and configuration of environments. This person will employ huge spreadsheets detailing the dependencies and configurations associated with each phase. In many cases, this can be a complex, error prone, bureaucratic process – and one that is getting worse as systems increase in complexity and number of integration points.

When the rare decision is taken to provision a new environment, often an entire funded project needs to be put into place in order to secure access to servers and engineering time. Environment provisioning is a long way from business as usual!

ENVIRONMENTS ARE EXPENSIVE TO MAINTAIN

Though virtualisation technology is mature, many organizations are still either running on physical servers or not taking full advantage of virtualization in order to achieve agility in their infrastructure. This means that environments become long lived, static and physical rather than short lived, agile and virtual.

Environments are also time consuming and manual to provision and manage, meaning that each environment introduced has a lot of human capital cost - usually expensive, skilled, busy engineers.

Within each environment, there may be license fees associated with external software such as monitoring systems or application servers. This further adds to the expense of creating and running new environments or building environments that are sized in line with production.

All of these cost implications incentivise organisations to keep as few environments as possible, increasing The Environment Logjam still further and continuing to introduce delay into the software development lifecycle.

THE ENVIRONMENT LOGJAM LEADS TO BIGGER, RISKIER RELEASES

Because environments are such a scarce resource that are hard to reconfigure and manage, people want to cram as much as possible into each release and go through the pain of moving releases through the environment pipeline as rarely as possible. As a result, each software release becomes bigger, more complex and riskier to deploy.

This is the antithesis of agile software development and continuous delivery, which aims to deploy smaller batches of change, much earlier and much more frequently.

ENVIRONMENTS ARE INCONSISTENT AND NOT PRODUCTION REALISTIC

Environments are often out of line, with test environments not accurately representative of the production environment. Often they are sized and configured differently at the outset for practical or cost reasons, or differences creep in over time as the environments are maintained and manually managed.

Integration points are another key source of environment differences. Upstream or downstream connections to third parties are sometimes limited in number, meaning that only one or two environments can have these connections. Other environments are either not connected, or connected to manually stubbed services. If environments are integrated differently then they are inconsistent and tests that are carried out in the pipeline are potentially compromised.

Data is another common source of inconsistency between environments. For either practical, risk, or regulatory reasons, it is often not practical to bring accurate copies of production data back into earlier environments. This represents another key source of environment inconsistency.

In our experience, these kinds of inconsistencies are the leading cause of production instability and outages. This is because code is not tested in environments that are an accurate reflection of production and so defects slip through which would be caught within more production realistic environments.

ENVIRONMENTS ARE STATIC AND MANUALLY MANAGED

Environments are very hard to resize and reconfigure, provision and tear down. As a result, they become static entities which are hard to rebuild in a repeatable way.

Changes to these static environments are very manual in nature. If we want to request changes, we need to raise tickets, sometimes to different external teams in the business. Sometimes, these changes are not implemented in a consistent way, further contributing to the inconsistency problem described above. Often, these changes take a long while to implement by the implementing teams, meaning that application releases are delayed.

We need to get much faster and more agile in the way we provision and manage environments, with more ownership of those environments and the infrastructure within them sitting with the development or change function. Doing this would add lots of speed to the process and allow us to get valuable new features into the hands of users much earlier.

ENVIRONMENTS ARE BECOMING EVEN MORE COMPLEX AND MANUAL TO MANAGE

As software becomes more complex, the supporting environments are also growing. For instance, 5 years ago a website that might have run on one big physical server would now be horizontally scaled out to hundreds of cloud servers.

In an enterprise environment, applications also become very deeply integrated into the application landscape. This makes the integration challenges described above even more apparent.

As mentioned above, because people are busy with new development and business as usual, the inefficiencies presented above tend to persist and become accepted as the norm. Individually they may create small pain points for teams in an ad-hoc manner, but when you step back and evaluate them together, they add up to a significant source of cost, delay and quality problems across the software delivery lifecycle.

The problem is also getting worse. The velocity of change is increasing as the business expect more and software and IT operations increasingly becomes a source of competitive advantage. Software and its associated infrastructure and environments continue to grow in complexity and new areas such as mobile software development are meaning that the landscape is ever changing. Unblocking The Environments Logjam is getting to the point where it really needs attention.

UNBLOCKING THE ENVIRONMENT LOGJAM WITH A DEVOPS APPROACH

The good news is that by leveraging DevOps approaches we can unblock The Environment Logjam, dramatically improve this situation, and lock in very real benefits as a result. In most of the organisations that we have seen, we believe that investing in this area would be the biggest step they could take towards reducing delivery cycle times, driving up efficiency and improving quality.

Below are some of the initiatives that you could put into place to achieve this.

MATURE USE OF VIRTUALISATION AND CLOUD

Mature use of server virtualisation allows us to provision, reconfigure and tear down environments as needed. For instance, when additional releases are flowing through the pipeline or when we decide to add a new testing phase, virtualization allows us to create new, short-lived, transient environments without application teams being blocked for access.

Cloud platforms take this one step further, giving us even more scope for automating server orchestration and offering almost unlimited elasticity and scalability without any capital costs.

Obviously this levl of infrastructure agility requires a degree of investment in modern virtualization platforms and the automation on top of those, but with servers being the building blocks of environments, having control and flexibility in server provisioning is a key tool in removing environment access as a source of delay.

MOVE TOWARDS INFRASTRUCTURE AS CODE

Infrastructure automation tools such as Puppet and Chef allow you to define infrastructure and configuration as source controlled code. Rather than, for instance, configuring a web server, database or application server by hand or a console, we will instead define these in source code artifacts which represent our infrastructure and configuration.

By managing infrastructure in this way, it means that building environments becomes much faster, more accurate, and more repeatable, ultimately giving us environments that are much more consistent with production. As well as unblocking environments as a source of delay, this reduces defects that slip into production due to inconsistencies in the environments.

This is particularly important in the short lived, transient environments described above where it is simply not practical to build such environments quickly and in a repeatable way by hand.

IMPLEMENT SERVICE VIRTUALISATION TOOLING

If it was just about creating and managing virtualized servers, many organisations would have put initiatives into place to unblock The Environment Logjam.

However, the real engineering complexity comes from the dependencies within environments – the upstream, downstream and external systems that exchange messages with the application under test.

Sometimes, access to those third party systems is limited, so that only one or two environments can be integrated into them in a production realistic manner. Even where access is available, they are sometimes out of the organisations control or subject to ongoing change, meaning that testing against those endpoints consistently is difficult.

Many organisations target this source of friction by designing manual stub services. Sometimes these stubs are brought up as part of the continuous integration and build phase in order to run tests against them, or perhaps these stubs are separate processes deployed into environments. However, these stubs tend to require a lot of investment and are often a lacking substitute for realistic integration testingendpoints in terms of flexibility and production realism. As a result, we tend to stick to integration testing in the one or two pinch-point environments with real connections, contributing to The Environment Logjam.

Implementing Service Virtualisation tools improves this situation, allowing you to build complex production-realistic stub services with less development effort. This

aids testing, making it more consistent and allows the organization to perform thorough integration testing earlier in the development cycle. These tools also open up environments so that they all become much more usable for integration testing, and are therefore a key tool in unblocking The Environment Logjam.

VIRTUALISE DATABASES

As with integrated services, data is also often a key sticking point and contributor to The Environment Logjam. Databases are big and copying them around takes a lot of time and storage costs. Often, database copies are restricted for regulatory reasons stemming from data security concerns. Typically, this process has to be carried out by a DBA, meaning delay and manual work for testers and the people tasked with getting applications to market.

As with servers and services, we also want our data to be aligned with production if we want to test in environments that are representative of production. If data is not realistic then testing is compromised and more defects are likely to slip through into production as a result.

Implementing Database Virtualisation tools allows us to combat data as a source of delay and inconsistency. These tools allow much more agility in how databases are managed and presented to environments, and can automate and facilitate self service for databases in the same way we might do with self service server provisioning or application deployments.

Database virtualization decreases cycle, removes delay from the process, removes manual work and improves consistency across environments.

SPEED UP THE PIPELINE WITH RELEASE AUTOMATION

As with any bottleneck, a key way to unblock The Environment Logjam is simply to move things through the phases much faster – particularly the slower phases of the pipeline.

Often, these stages are associated with testing – system test, system integration test, performance testing and user acceptance testing environments for instance.

Therefore, more investment in test automation can be a simple but effective lever in unblocking environments and reducing cycle time.

Releasing software into environments should also be made faster, more reliable and moved towards a self service model. This means that development and test teams are not left waiting for other people to carry out tasks on their behalf and there are less failed deployments introducing delay. Again, small savings in problems like this can add up to significant improvements in end to end cycle time.

Environments and test phases should be arranged in the most optimal way too. This means that the test phases should be arranged in parallel or with the shorter lived, highest impact tests being ran first and in isolation. This means that quality is 'shifted left' and that we aren't carrying out tests on release candidates, only to have them abandoned further down the release pipeline.

PEOPLE & PROCESS

Unblocking The Environment Logjam is not all about automation and tooling. Sometimes, changing business processes and working practices are a much more effective means of doing this before even a line of coding is written.

One example of this might be moving responsibilities within your IT organization. Allowing testers to selfservice deploy applications or developers to own and manage more of their infrastructure are two examples of this that reduce handovers and delay and speed up end to end delivery cycle time.

At a higher level, simply moving towards better collaboration between technology stakeholders and engineering teams will also contribute to unblocking The Environment Logjam. Sometimes, something as simple as people picking up the phone to a colleague and making a small adjustment to your plans can avoid significant bottleneck situations.

Another example which we preach often is 'shifting left' - having testers and operations people involved in the design and development of the software. This can lead to software flowing through release pipelines much reliable and predictably.

THE BUSINESS CASE

The Environment Logjam is very real, and the biggest single cause of delay and quality issues that we have observed amongst our enterprise clients. DevOps inspired automation and working practices will help to this allow significantly reduced cycle times of higher quality software. Considering this, here is what we believe to be the business case for investing in an improvement initiative in this area.

GET TO MARKET FASTER

By unblocking The Environments Logjam, you'll be able to get products to market earlier and iterate on them more frequently. This means that valuable software releases are in the hands of users rather than being held up waiting for environments or moving through slow moving delivery pipeline phases.

IMPROVE EFFICIENCY

By putting initiatives such as the above into place, there will be less time wasted waiting for environments and doing repetitive, manual work around environments and application deployments. This allows engineering resources to concentrate on truly value creating activities.

IMPROVE QUALITY

By virtualizing platforms and systems and managing them with more consistency, our path to production becomes much more consistent and repeatable. Test environments look more like production and so we test in environments that are accurately representative of the production system in terms of application code, integration points and data.

MAKE AGILE DEVELOPMENT PAY

Over the last decade, many organisations have adopted agile ways of working. However, what we frequently see is that this simply means the bottleneck has been moved downstream to later phases in the software development lifecycle. We believe that by investing in environment management and release automation, we can finally start to leverage some of the benefits of agile ways of working.

COST SAVINGS & REVENUE BENEFITS

Though an environment automation initiative will cost money, we believe it will generate a return on investment.

At a high level of maturity your IT organisation can spin up test environments that are an exact replica of production, carry out the testing, and then tear those environments down. This is likely to yield a net saving in infrastructure costs over static, physical environments.

At the same time, with applications releases flowing through into the hands of users earlier and more often, the business benefits through increased revenue associated with those features.

ACHIEVE COMPETITIVE ADVANTAGE

Whilst you are achieving the benefits of more infrastructure agility, reduced delivery cycle times, improved quality and better software in the market, your competitors will be waiting for access to their test environments.

ABOUT CONTINO

Contino are a boutique consultancy that specialises in helping organisations adopt DevOps and Continuous Delivery tools, practices and approaches. Contino work with organisations to enable them to deliver software faster and more efficiently whilst retaining quality. For more information, visit contino.co.uk or follow us at @continouk.

Contact Us:

Benjamin Wootton
+447463 898 809
Benjamin.Wootton@contino.co.uk

Matt Farmer
+447903 843 529
Matt.Farmer@contino.co.uk